Mariana Ortega Ramírez
Alejandro Lancheros Duarte

Proyecto 2: Documento de Diseño

1. Contexto del Problema:

Nuestro sistema hace dos tres cosas principales: Mostrar información referente a la empresa de Alquiler de Autos, es decir mostrar las Categorías Disponibles dentro del sistema y las condiciones básicas que debe cumplir cada categoría. Estas condiciones básicas se definieron ya que una persona si no hay disponibilidad en la categoría que desea, se le ofrece otra, pero está debe seguir lo mínimo que requiere una categoría; adicionalmente debe realizar las funcionalidades de cada uno de los tipos de usuarios. Para el cliente debe ser posible registrarse y realizar una reserva. El administrador puede agregar un vehículo, dar de baja un vehículo, registrar a un empleado y configurar nuevos seguros. El administrador de una sede especifica debe poder cambiar la sede de un auto. Un empleado debe poder ser capaz de Entregar un vehículo, y recibir algún vehículo; por último, la página debe lograr verificar los login y contraseña de cada uno de los usuarios.

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

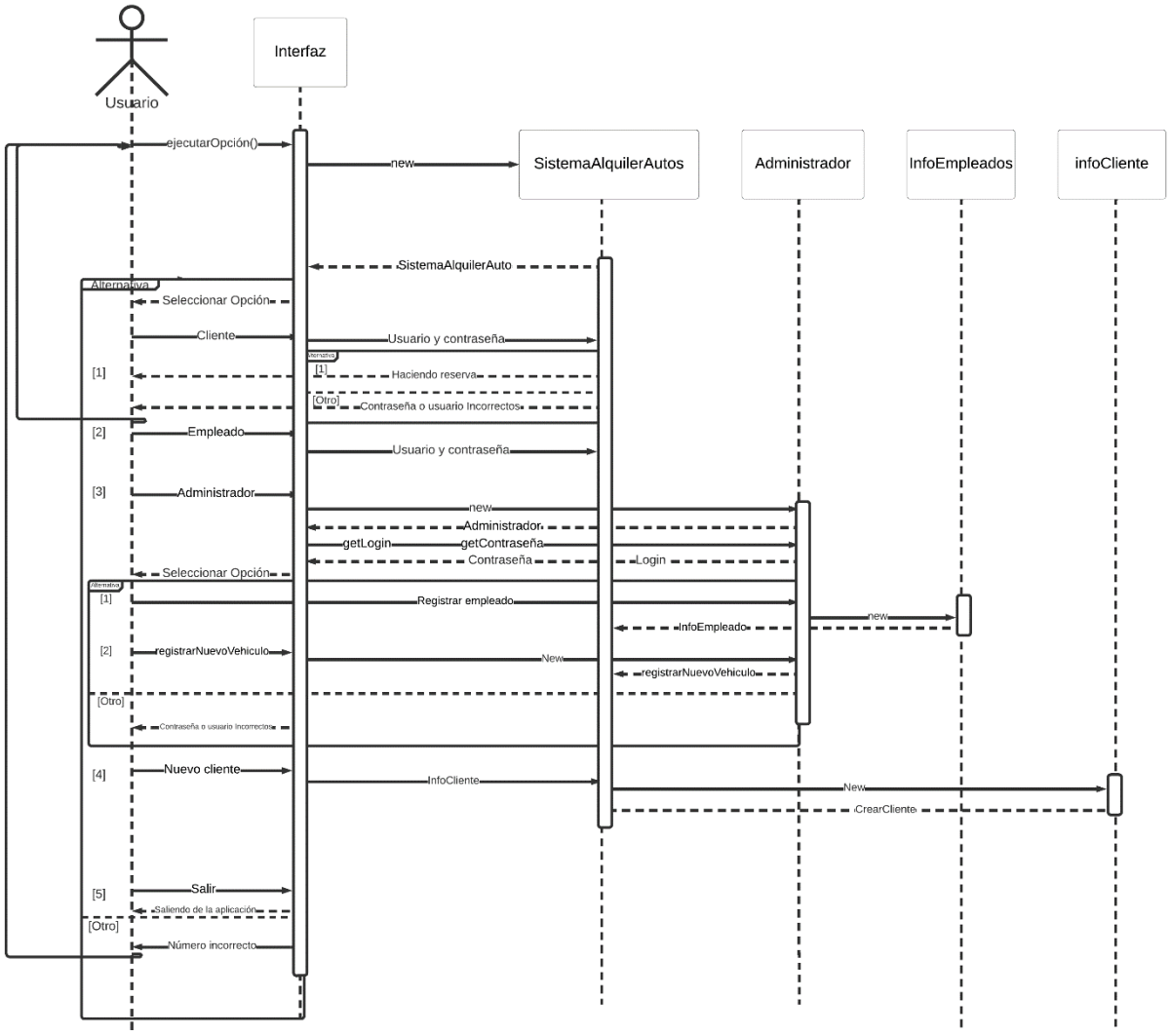

Figura 1

Ejemplifica como seria cada secuencia según el usuario que ingresa al sistema, para cada uno de ellos si alguno de los parámetros ingresados (Usuario y Contraseña) es invalido, este debe volver a ingresarlos.

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

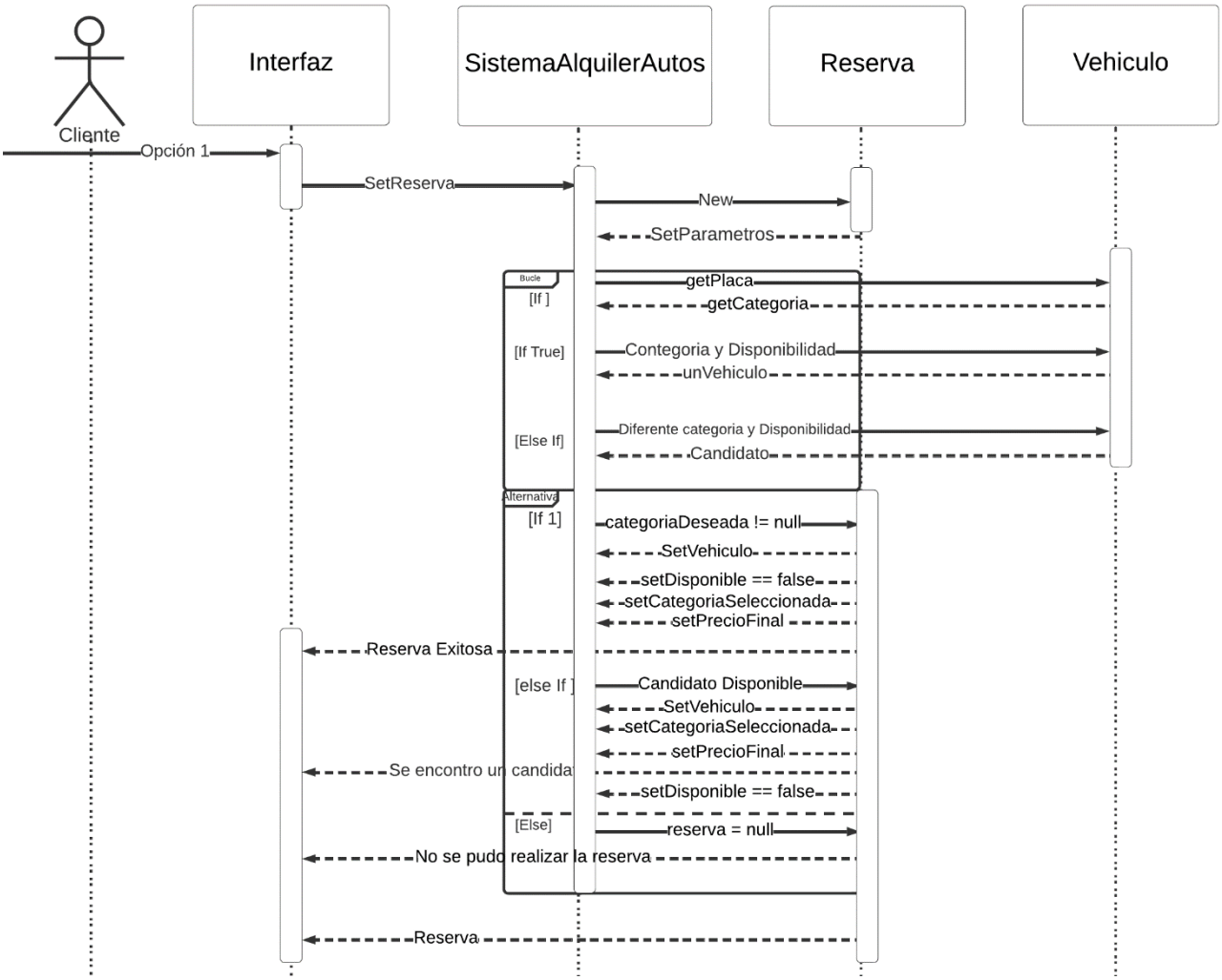

Figura 2

Este diagrama va de la mano con el anterior, debido a que demuestra cómo se lleva a cabo la acción de realizar una reserva por parte del cliente. Existen dos posibilidades, el cliente encuentra un vehiculo de la categoría correcta si está disponible, el sistema realiza la reserva, si no hay un vehiculo de la categoría correcta, encuentra un candidato que cumpla con las necesidades del cliente y se lo asigna.

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

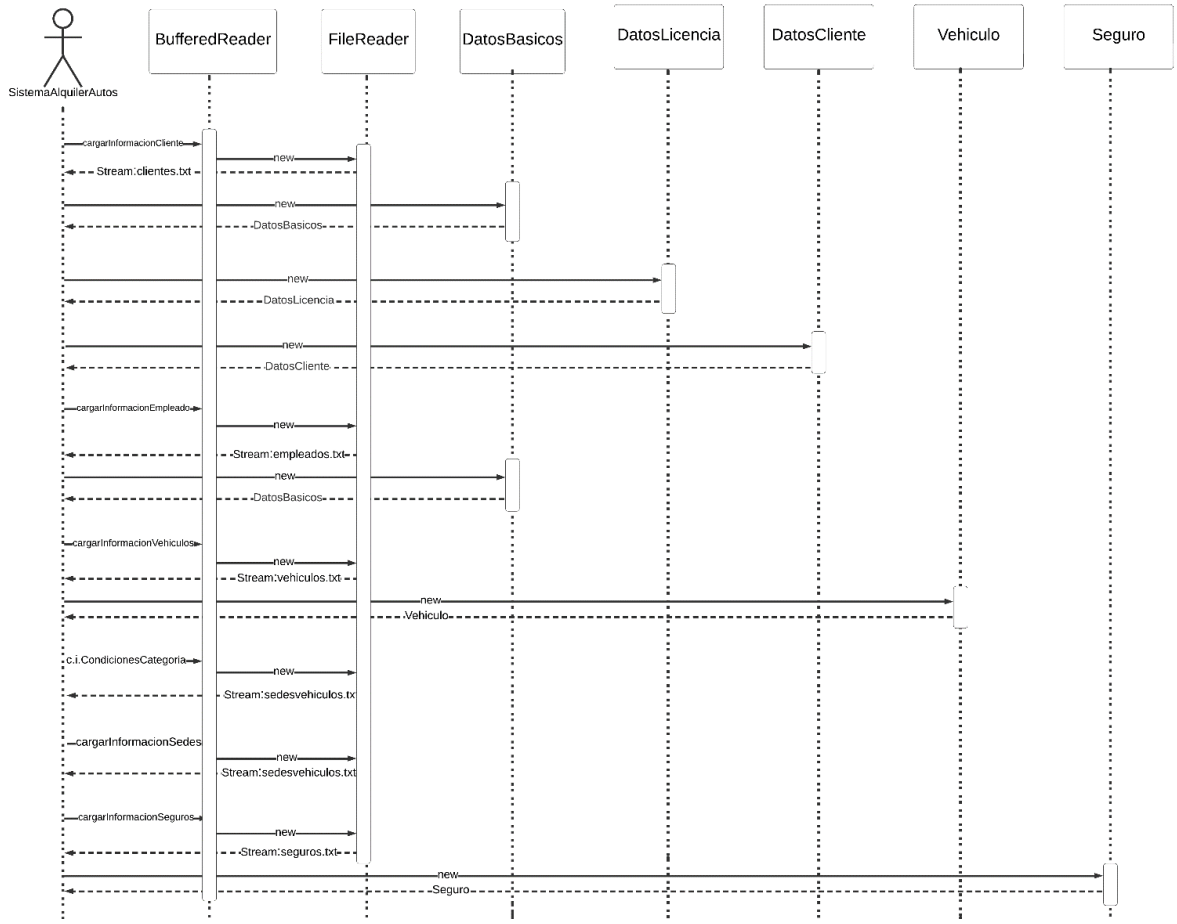

Figura 3

Aquí se puede ver como se lleva a cabo la carga de los datos y la actualización de clientes y empleados, toda esta información va a terminar siendo almacenada por el sistema de alquiler de autos.

Anteriormente se había utilizado la clase dentro de consola Interfaz, la cual es una interfaz por consola. Sin embarga para esta entrega se implementó una interfaz grafica por medio de Java Swing y Java awt.

2. Componentes Candidatos Y Estereotipos

1. Reconocemos que debe existir un componente encargado de manejar toda la información referente a las sedes, categorías, información del cliente, información de los empleados, información administradora general, los administradores de cada una de las sedes, etc, el componente seleccionado es **CarpetaArchivos** y será nuestro **I*nformación Holder***.

2. En relación con la transformación y difusión de la información existente se nos ocurrió asignar el componente **interfaz** como nuestro ***Interfacer***

Este rol se encarga de ejecutar el main del programa, adicionalmente de mostrarle al quien usa el sistema las opciones que tiene. Este componente se encarga de dar los accesos

dependiendo del tipo de usuario, y así darle acceso a lo que puede realizar dentro del sistema.

3. A razón de que el **administrador general** tiene la posibilidad de registrar nuevos vehículos y dar de baja los mismos, además de encargarse de la configuración de los seguros, los cuales implican un costo adicional por día, le asignamos el estereotipo ***Cordinator***.

4. A causa de que necesitamos manejar gran cantidad de información relacionada con la ubicación y disponibilidad de cada vehículo para ser alquilado, la creación de un historial de reservas y modificación de las reservas, además de calcular tarifas, asignamos al macro componente **sistema** como ***controller.***

El rol de SistemaAlquilerAutos es el que realiza mayoría de interacciones y comunica a los componentes del proyecto. Adicionalmente acá se manejan los datos y se cargan los txt ya existentes y adicionalmente se agrega información e instancias a estos. De igual manera dentro de este rol se guardan la mayoría de las estructuras para organizar el proyecto y ejecutar los métodos desde ArrayList hasta mapas.

5. Sin embargo, sistema no es suficiente para cumplir con toda la prestación de servicios que necesitan los demás componentes, aun es necesario en este caso registrar a los clientes, iniciar la reserva, ofrecer los seguros, definir la tarifa diaria, entregar y recibir los vehículos, etc. De esto se encargará el macro componente **empresa** y también desempeñará el rol de ***service provider.***

La figura 4 presenta los cinco macro componentes candidatos con su respectivo estereotipo.

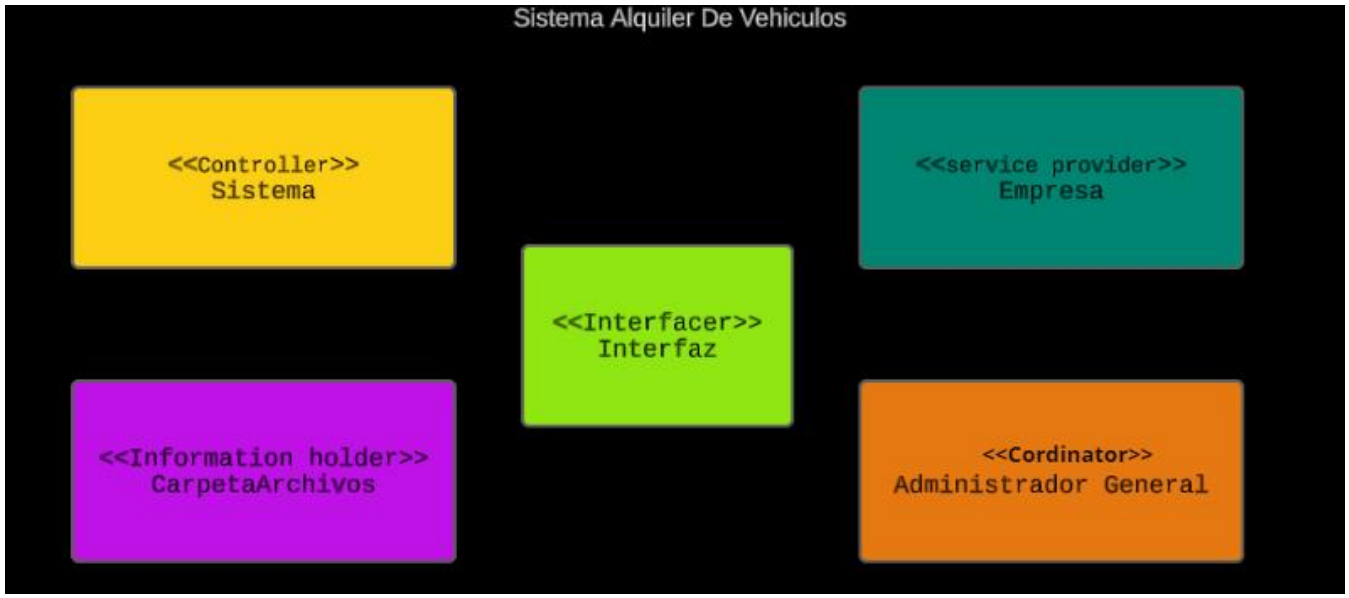

Figura 4. Componentes candidatos y sus estereotipos

3. Explicación Decisiones de Diseño y explicación componentes:

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

Para la implementación del proyecto se adaptó su código para que siguiera el patrón de diseño MCV (Modelo, Vista, Controlador). Dentro del paquete alquilerAutos.modelo, se guardan todas las clases relacionada con la lógica del problema. Las clases que albergan el **Modelo** son:

- Administrador
- AdministradorSede
- Categoria
- DatosBasicos
- DatosCliente
- DatosLicencia
- Empleado
- InfoClientejava
- InfoEmpleado java
- OpcionesCategoriajava
- Sede
- Seguro
- Vehículo

Cada una de las clases contiene los métodos que les permiten ejecutar los requerimientos funcionales de la aplicación. Siguiendo lo establecido por el patrón MVC, acá se establece toda la lógica.

Por otro lado, declaramos nuestro **Controlador** del MVC, como sistema. Dentro de este paquete, se encuentra la clase de SistemaAlquilerAutos, la cual tiene los siguientes atributos:

- ArrayList<Reserva> reservas
- ArrayList<Vehiculo> vehiculos
- ArrayList<DatosCliente> clientes
- ArrayList<DatosBasicos> empleados
- ArrayList<Seguro> seguros
- Map<String, ArrayList<String>> sedes. Esto se hace como un hashMap que tiene como “llave” es el nombre de una sede, y el “valor asociado”, es un arreglo con las placas de los vehículos que tiene esa sede.
- Map<String, String[]> condicionesCategoria. Esto lo que tiene es un HasMap, que tiene como “llave” el nombre de una categoria, y como “valor asociado” tiene un arreglo de String, que son las condiciones básicas que debe cumplir la categoría.
- ArrayList<AdministradorSede> adminsedes

Por otro lado, los métodos que posee esta clase, que son lo que conectan la vista (la interfaz gráfica) con el modelo de la aplicación. Los métodos que ayudan a mantener la persistencia del sistema son todos los que inician con CargarInformacion. Los métodos que inician con verificar son los que se utilizan para verificar si el login y contraseña que se pasan en la

interfaz gráfica son válidos para cada tipo de usuario. El resto de los métodos son variados ya que corresponden al resto de requerimientos para cada uno de los componentes del modelo.

- buscarVehiculo(Vehiculo): String
- calucularTarifas(String, Vehiculo, int) : int
- cambiarVehiculoSede(String,String):boolean
- cargarAdminSedes(): void
- cargarInformacionCliente(): void
- cargarinformacionCondicionesCategoria(): void
- cargarinformacionEmpleado(): void
- cargarInformacionReservas() : void
- cargarInformacionSedes(): void
- cargarInformacionSeguros(): void
- cargarinformacionVehiculos(): void
- crearReserva(String, String, String, String, String, String, String, String) : Reserva
- diasReserva(String, String) : int disponibilidad Vehiculo (Vehiculo) : String
- eliminarVehiculo (String) : boolean
- entregaVehiculo (String) : boolean
- getDatosCliente (String) : DatosCliente
- getDatosReserva(Reserva) : ArrayList<String»
- input(String) : String.
- modificarFechaEntrega(String): void
- modificarRangoEntrega(String):void
- modificarSedeEntrega(String): void
- nuevoCliente(String, String, String, String, String, String, String, String, String) : boolean
- nuevoEmpleado(String, String, String, String, String, String) : boolean
- nuevoSeguro (String, String, String) : boolean
- ofrecerSeguro(): void
- printSede(): void
- recibirVehiculo(String) : boolean
- registrarAdminSede(String, String, String, String, String) : void
- registrarNuevoVehiculo(String, String, String, String, String, String, String, String, String, String, String) : boolean
- setReserva(String, Reserva) : void
- verificarAdministrador(String,String):boolean
- verificarAdminSede (String, String) : boolean
- verificarCliente (String, String) : boolean
- verificarEmpleado(String, String) : boolean

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

Finalmente, a diferencia del Proyecto 1, el paquete de consola, la cual es nuestra **Vista** dentro del patrón de diseño, no solo tiene la clase Interfaz, la cual es la encargada de hacer la interfaz por consola. Ahora se tienen muchas clases más clases que representan la creación de la interfaz gráfica. La clase principal que ejecuta la interfaz gráfica es PanelTabb. Este panel lo que implementa un JTabbedPane. Esto se decidió de esta manera ya facilitaba la implementación de los distintos tipos de funciones que debía cumplir la aplicación. El JTabbedPane se maneja como si fuera una ventana, pero son diversas pestañas pasando de manera simultánea.

Como primera pestaña "About Us", es la que le muestra al usuario la información básica de todo el sistema de alquiler de autos. Como las categorías que posee la empresa con sus condiciones básicas y las sedes que tiene la empresa, junto con las categorías que hay disponibles en cada sede. Estas para llamar la atención de posibles clientes, y en el momento de un cliente estar haciendo una reserva pueda volver a esta pestaña a revisar la información de las sedes y las categorías.

Las siguientes pestañas son de inicio de sesión para cada uno de los usuarios que tiene el sistema. Esta decisión de diseño se dio para facilitar la verificación de los login y contraseña. De igual manera evita confusiones en caso de que dos personas de distinto tipo de usuario tengan el mismo login, y le de ingreso al que es. En caso de que la verificación falle, se muestra un JOptionPane que le pide al usuario que vuelva a intentar iniciar sesión. Se utilizó un JOptionPane para que en el caso de que el usuario por accidente no ingresó correctamente los datos pueda reingresarlos sin tener que reiniciar el programa.

La primera de las pestañas de inicio de sesión es la del cliente. El panel que se muestra inicialmente es el inicio de sesión, que por medio de un GridLayout se organizan Los JLabels que le piden el username y la contraseña al usuario, los JTextField para que el cliente ingrese su información, y un botón de ingresar, el cual por medio de un ActionListener revisa si es correcto, para ver esto llama al controlador y verifica si es correcto. Si lo es, por medio de un CardLayout, muestra el que corresponde al panel de contenido de lo que puede hacer el cliente en la aplicación. El panel de funciones del cliente están 3 botones, realizar reserva, modificar reserva (no hace nada) y registrar cliente (en el primero documento explicando el proyecto no se especificó quien registraba a un nuevo cliente, así que, dentro de nuestra lógica para registrar a un nuevo cliente, se necesita que lo registre otro cliente). El primer botón, por medio de JLabels le pide al usuario los datos para una reserva, y por medio de JTextField para datos que pueden ingresarse más "libremente", y JComboBox para cosas que no queremos que el usuario realice errores al escribir la sede o la categoria. En el caso de que se realiza se le muestra la información de la reserva al usuario, en caso contrario le dice que intente hacer de nuevo la reserva porque se generó un error. Los posibles errores es que no hay vehículos disponibles para una sede, o una categoria, o no hay una alternativa de otra clase que cumpla los mínimos de la clase pedida por el usuario. En el botón de registrar usuario se le pide al usuario su información básica y los datos de su licencia, y si se registra se muestra un mensaje de registro exitoso. Para realizar cada uno de los botones se crea un panel y algunos paneles auxiliares, en

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

Cliente se crean PanelRealizarReserva con dos paneles para este panel que son ReservaCorrecta y ReservaIncorrecta, y PanelRegistrarCliente.

La siguiente pestaña es la del empleado. El panel que se muestra inicialmente es el inicio de sesión, que por medio de un GridLayout se organizan Los JLabels que le piden el username y la contraseña al empleado, los JTextField para que el empleado ingrese su información, y un botón de ingresar, el cual por medio de un ActionListener revisa si es correcto, para ver esto llama al controlador y verifica si es correcto. Si lo es, por medio de un CardLayout, muestra el PanelEmpleadoInicial que corresponde al panel de contenido de lo que puede hacer el empleado en la aplicación. El panel de funciones del cliente están 2 botones, entregar vehículo, y recibir vehículo. En ambas opciones el empleado ingresa la placa del vehículo, en el caso de entregar, cambia la disponibilidad de ese vehículo a false, y en recibir lo cambia a true. Para realizar cada uno de los botones se crea un panel y algunos paneles auxiliares, en Empleado se crean PanelEntregarVehiculo y PanelRecibirVehiculo.

Posteriormente está la pestaña para el Administrador que es solo uno. El panel que se muestra inicialmente es el inicio de sesión, que por medio de un GridLayout se organizan Los JLabels que le piden el username y la contraseña al Administrador, los JTextField para que el administrador ingrese su información, y un botón de ingresar, el cual por medio de un ActionListener revisa si es correcto, para ver esto llama al controlador y verifica si es correcto. Si lo es, por medio de un CardLayout, muestra el panel que corresponde al PanelAdminInicial de contenido de lo que puede hacer el administrador en la aplicación. El panel de funciones del administrador son 4 botones: Eliminar Vehículo, Agregar Vehículo, Agregar Empleado, Configurar Seguro. Los botones que se encargan de Agregar o Configurar, por medio de JLabels se le dice al administrador que datos debe ingresar, y por medio de JTextField el administrador puede ingresar los datos pertinentes. Mientras que para Eliminar un Vehículo lo que hace es pedirle por medio de JLabel y JTextField la placa del vehículo que se quiere eliminar, y esto simplemente invoca métodos del sistema, que procuran mantener la persistencia. Para realizar cada uno de los botones se crea un panel y algunos paneles auxiliares, en Administrador se crean los paneles PanelConfigurarSeguro, PanelAgregarVehiculo, PanelEliminarVehiculo y PanelAgregarEmpleado.

Por ultimo se encuentra la pestaña de los Administradores de cada sede. El panel que se muestra inicialmente es el inicio de sesión, que por medio de un GridLayout se organizan Los JLabels que le piden el username y la contraseña al Administrador de sede, los JTextField para que el administrador de sede ingrese su información, y un botón de ingresar, el cual por medio de un ActionListener revisa si es correcto, para ver esto llama al controlador y verifica si es correcto. Si lo es, por medio de un CardLayout, muestra el PanelAdminSedeInicial que corresponde al panel de contenido de lo que puede hacer el administrador de sede en la aplicación. El panel de funciones del administrador de sede se reduce a la función al botón de cambiar sede en caso de una reserva especial. Para esto por medio de JLabels y JTextFields le pide al administrador de la sede la placa del vehículo del cual se desea cambiar la sede, y la sede a la cual se cambiará, para esto simplemente invoca al controlador con un método que ejecuta esto mismo. Para realizar cada uno de los botones

se crea un panel y algunos paneles auxiliares, en el caso de Administrador sede PanelCambiarSede.

4. RESPONSABILIDADES

A continuación, entramos más en detalle en cada una de las responsabilidades que ejecutara cada uno de los macro componentes.

Tabla 1: Asignación de responsabilidades

#	RESPONSABILIDAD	COMPONENTE
1	Calcular tarifas	Sistema
2	Informar donde está vehículo (sede)	
3	Informar disponibilidad de vehículo especifico	
4	Seleccionar auto para cliente	
5	Crear formato historial reserva	
6	Modificar reserva	
7	Puede reservar auto	
8	Registrar nuevos vehículos	Administrador General
9	Dar de baja vehículo	
10	Configura seguros	
11	Registrar empleados	
12	Registrar cliente	Empresa
13	Cobrar 30% alquiler	
14	Definir tarifa diaria por categoría	
16	Ofrecer seguros	
17	Entregar auto (cambiar disponibilidad)	
18	Recibir auto (cambiar disponibilidad)	
19	Cargar reserva a los archivos	CarpetaArchivos
20	Guardar información cliente	
21	Crear nuevo vehículo	
22	Guardar nuevo vehículo	
23	Crear cliente	
24	Guardar cliente	
25	Crear empleado	
26	Guardar empleado	
27	Pedir login y contraseña	Interfaz
28	Mostrar opciones para cliente	
29	Mostrar opciones para empleado	
30	Mostrar opciones para administrador	
31	Ejecutar opciones	

Mariana Ortega Ramírez
Alejandro Lancheros Duarte

5. Colaboraciones

Inicialmente hay que tomar en cuenta que el estilo de control de este proyecto es delegado, es decir la Interfaz le indica al sistema que debe hacer, y el sistema les dice a los demás componentes que deben hacer para ejecutar cada una de las opciones.

Entre las responsabilidades antes mencionadas también hemos planteado algunas colaboraciones entre los componentes para cumplir con funcionalidades más amplias.

1. La interfaz--->Sistema. se conecta para cargar información. De la interfaz se conecta al Sistema y le pide cargar la información de los txt para cada uno de los atributos que tiene SistemaAlquilerAutos.
2. Interfaz -- > Sistema --> CarpetaArchivos -- > Sistema: Esta colaboración se da cuando se desea crear un nuevo cliente., La interfaz ejecuta la opción, el sistema utiliza el método que llama a la clase de InfoClientes y DatosCliente que hace parte del information holder. . Al final vuelve al sistema para agregarlo a el arreglo de clientes
3. Interfaz -- > Sistema --> Administrador -- > CarpetaArchivos->Sistema: Esta colaboración se da cuando se desea crear un nuevo empleado., La interfaz ejecuta la opción, el sistema utiliza el método que se comunica e inicializa el administrador el cual ejecuta el método de registrar el método de registrar a un empleado, llamando a los datos y clases que se encuentran en la CarpetaArchivos como InfoEmpleado y DatosBasicos. Al final vuelve al sistema para agregarlo a el arreglo de empleados
4. Interfaz -- > Sistema --> Administrador -- > CarpetaArchivos -- > Sistema: Esta colaboración se da cuando se desea crear un nuevo seguro., La interfaz ejecuta la opción, el sistema utiliza el método que se comunica e inicializa el administrador el cual ejecuta el método de configurar un seguro, el cual se va a la carpeta de archivos donde está guardada la clase seguro. Al final vuelve al sistema para agregarlo a el arreglo de seguros
5. Interfaz -- > Sistema --> Administrador -- > CarpetaArchivos -- > Sistema: Esta colaboración se da cuando se desea crear un nuevo vehiculo., La interfaz ejecuta la opción, el sistema utiliza el método que se comunica e inicializa el administrador el cual ejecuta el método de registrar un vehículo, el cual se va a la carpeta de archivos donde está guardada la clase de vehículo. Al final vuelve al sistema para agregarlo al mapa que está organizado como llave una sede y como valor asociado un arreglo con todos los vehículos en la sede.
6. Interfaz -- > Sistema -- > CarpetaArchivos. Esta colaboración se encarga de comunicar que la interfaz se pidió hacer una reserva, el sistema por medio del método le pide al usuario los datos necesarios. Pero este se debe comunicar con la CarpetaArchivos, específicamente con vehículo, ya que en este existe el método de verificar condiciones el cual es necesario cuando no existe un vehículo con la categoría deseada del usuario. Le da lo que se acomode a sus necesidades (Eso se estableció así ya que, si por ejemplo una persona reserva para un miniván que tiene como requisito principal que tenga 7 o más puestos, no se le ofrecerá la siguiente categoría que sería un vehículo de lujo que tengo 4 puestos).

Mariana Ortega Ramírez
Alejandro Lancheros Duarte